Information Technology Industry Council
Leading Policy for the Innovation Economy

July 8, 2011

Ms. Marlene Dortch
Secretary
Federal Communications Commission
445 Twelfth Street, S.W.
Washington, D.C. 20554

Re: Ex parte meeting on CG No. 10-213, WT No. 96-198, CG No. 10-145

On June 10, the Information Technology Industry Council ("ITI") filed an *ex parte* report with the Federal Communications Commission (hereafter, "Commission") regarding a previous meeting that we had held with staff regarding the Commission's Notice of Proposed Rulemaking implementing the Twenty-First Century Communications and Video Accessibility Act of 2010 ("Accessibility Act"). The discussion revolved primarily around provisions related to Advanced Communications Services ("ACS").

During the meeting, staff commented that it would have been of beneficial had the Commission's technical experts been able to participate in the discussion. We expressed a willingness to schedule a second ex parte meeting in order to provide such an opportunity, and plan to request a second ex parte meeting on technical issues related to the implementation of the ACS provisions. To help facilitate a productive dialog, we are pleased to submit additional input.

Our comments follow:

1. **Manufacturers and developers, including platform and OS developers, should only be held accountable for that which is reasonably within their control.**

During the June 8 *ex parte* meeting with Commission staff, ITI emphasized the fact that many devices that deliver ACS consist of multiple components – hardware, operating system platforms, software applications, etc. – which in combination provide and enable the diverse functionality of such products. Each component in the "stack" is essential to the product's overall "ecosystem." Even so, manufacturers and service providers who develop the various components of an accessible ACS typically only have control over their own components. Accordingly, we urged the Commission to ensure that the regulations implementing the

Accessibility Act's ACS provisions limit manufacturer and service provider accountability solely to those components over which they have control.

ITI agrees with the FCC that that, in many cases, "ICT devices have evolved from being typically single-function devices into general purpose computers, with an architecture reflecting this evolution," and that the various components of the technology stack "may be created by a different manufacturer and sold separately." We believe, however, that there may be several components missing from the Commission's conceptualization: the assistive technology ("AT") utilized by the end user; a set of Accessibility Services provided by the operating system ("OS") (or platform); and for web-based ACS, the web browser.

To further illustrate the concept of the "stack," we have attached a document entitled "Model of ICT Accessibility Components," as well as a complementary textual description of the images contained in the document. The document contains a conceptual model of components that deliver ACS (image in the upper left corner of the document), along with five examples of various hypothetical device "ecosystems" containing bundled components that together enable ACS. Our hope is that we will be able to schedule another meeting with Commission policy and technical staff during which we will provide complementary demonstrations to illustrate the examples referenced below.

Examples one, two and three of the document, all would have the same "Access jQuery" application running, i.e., on a Mac with VoiceOver, then on Windows with NVDA, and finally again on Windows but with JAWS®. In each example, the ACS software is unchanged, yet the end user experience would be different, depending upon factors completely outside of the control of the ACS software manufacturer. In the first case, the user experiences the ACS with "built-in" accessibility – it just works with the accessibility that is built into the Macintosh computer itself. In the second case, the user experiences the ACS with "nominal cost" accessibility, i.e., a free NVDA screen reader that the end user would install on the Windows-based computer. In the third case, the user experiences the ACS through compatibility with the 3rd party commercial AT, i.e., JAWS. These are the three distinctions made in the Accessibility Act and proposed regulations, yet achieving one or the other of these three contemplated outcomes occurred *without any difference in the behavior of the ACS manufacturer*.

Again, in the case of ACS software, all that a manufacturer or service provider can control is their own software; not the platform or OS, not whether the OS comes with built-in accessibility features, not whether there exists nominal cost AT, etc. In addition, for any given OS, solutions may come on the market after the ACS ships (e.g., Mac OS v10.2 did not include VoiceOver, but v10.4 did, instantly making many ACS applications accessible via built-in AT.

During our June 8 *ex parte* meeting, Mr. Tom Wlodkowski of AOL provided a demonstration of a native mobile application demonstrating that the model ITI proposes applies equally well to ACS on high-end mobile or smart phones. He demonstrated AOL's AIM chat application for iPhone working with VoiceOver. The ACS application supports built-in accessibility in the same way that the Access jQuery does, by supporting the appropriate accessibility services for

the platform on which it is running. The key, again, is that the platform has built-in AT available along with a set of accessibility services that ACS applications can support in order to be accessible with that AT. We will be happy to repeat this demonstration during a second meeting.

Finally, we wish to note that, where ACS accessibility requirements address matters like hearing aid compatibility or decibel volume gain on a speaker, ACS software cannot overcome hardware limitations. In many cases the hardware is sold for a broad range of purposes without any included ACS software that would trigger the ACS hardware requirements.

2. The Commission should limit the definition of "existing" peripheral devices and customer premises equipment to those that are currently sold, rather than include devices and equipment that might still be used but that are not currently sold.

As we described in #1 above, assistive technology is a key component of the "ICT stack" - the ICT version of what is called "specialized customer premises equipment" in Section 255 parlance. When the assistive technology is a 3rd party commercial software application, requiring that ACS software compatible with the assistive technology that a user already owns means the user wouldn't need to purchase something new. But this can be a challenge in the ICT environment, where new technologies emerge rapidly, and where often the only way to make such new technology accessible is through the use of a new set of Accessibility Services.

To illustrate the challenge, we note that while current versions of the Windows screen readers JAWS® and WindowEyes® support both the UI Automation and IAccessible2 accessibility services, earlier versions did not. Accordingly, manufacturers and service providers cannot utilize these accessibility services to make ACS accessible via older versions of the screen readers. We also note that once-popular AT products such as Slimware Windows Bridge have been discontinued and are no longer being supported by their creators, and will never interoperate with ACS software via newer accessibility services.

Thus, requiring that ACS software must work with *all* AT that *any* user has would effectively mean that the Accessibility Act must hinder ICT innovation. New user interface techniques and web technologies like HTML5 couldn't be used for ACS under the CVAA, as the new accessibility services they use – such as WAI-ARIA for HTML5 – are only now being integrated into AT in a consistent way.

Therefore it is critical that the Commission limit the definition of "existing" assistive technology to those that are currently available in the marketplace.

3. The Commission should phase in the requirements of the Accessibility Act, including enforcement provisions, over a period of time that is consistent with available technology and the manner in which ACS equipment and services are developed.

ICT components are developed on a variety of timetables. Some web-based products are updated continuously. Many consumer electronics hardware devices are rolled out in as little as 6-9 months. By comparison, major applications like browsers and office suites may take 1-3 years to develop, and operating systems may take 5 years or more. Within an ICT development timeline, most of the features and functionality are decided within around the first third of that cycle. For example, if a 3 year cycle for an ACS application or operating system began in September 2011, it might ship September 2013, nearly two years after the ACS regulations are published and in effect. It would most likely be too late to alter the product functions if such were necessary to meet the ACS regulations without causing a lengthy delay and incurring significant costs. A phased-in approach with give all manufacturers the opportunity to build into their processes necessary design changes, as well as work with other component manufacturers to achieve successful interactions that enhance accessibility.

Specifically, ITI recommends that the Commission delay enforcement for a minimum of two years, and consider exercising its general waiver authority for classes of products with development cycles that typically extend beyond two years, such as operating systems.

4. The Commission should recognize that in a number of cases enterprise software applications may be "Customized Equipment and Services" that are "not offered directly the public."

While much of the ACS ICT sold to enterprises is also sold in stores to the general public (e.g. personal computers, personal productivity software), there is a significant class of enterprise software applications that are never sold to the general public. Known as "Enterprise Software,"[1] it generally has the following characteristics:

- It is developed for enterprise- or organization- wide deployment
- It focuses on key aspects of running the enterprise and its business (e.g. general business functions such as Accounting, Supply Chain management, Customer Relationship Management, HR; industry-specific business such as Manufacturing, Retail, Healthcare)
- It is typically customized by the customer to their enterprise/organization, adapted to the specific way their business runs
- It is administered by the IT staff of the organization, not by individual departments, let alone individual employees
- It is commonly developed to meet Section 508, and to be compatible with assistive technologies (both built-in and 3rd party), though enterprise customization may override or bypass accessibility features created by the manufacturer

One example of enterprise software that may contain ACS is a Customer Relationship Management (CRM) suite. The basic business logic of CRM software comes from the software manufacturer, but that enterprise software is customized for each individual enterprise's IT

[1] For further information on use and characteristics, *see* http://en.wikipedia.org/wiki/Enterprise_software

environment – adapted to the enterprise's product and customer databases, their product defect tracking system, their troubleshooting guides and knowledge base of known workarounds, their public website, etc.

CRM suites have started including ACS in the form of a live web-chat module, which enables a customer to communicate with a live person through a chat window. Typically customers can get the live chat only after first reviewing troubleshooting information and knowledge base articles relevant to the product(s) they own. The particular user interface for this chat session is customized for the enterprise: the enterprise employee serving the customer has an enriched chat user interface displaying all of the troubleshooting material and knowledge base articles that each customer they are chatting with has reviewed, and the answers to all of the on-line questions they already completed.

Enterprise software customization, such as is done for CRM suites, may introduce accessibility issues as the enterprise customizes the user interface and operation of the software to operate in their IT environment. The original software manufacturer should not be held responsible for the accessibility of such customized enterprise software, which is keeping with Congress' intent when it created the exemption for customized equipment and services in 716(i).

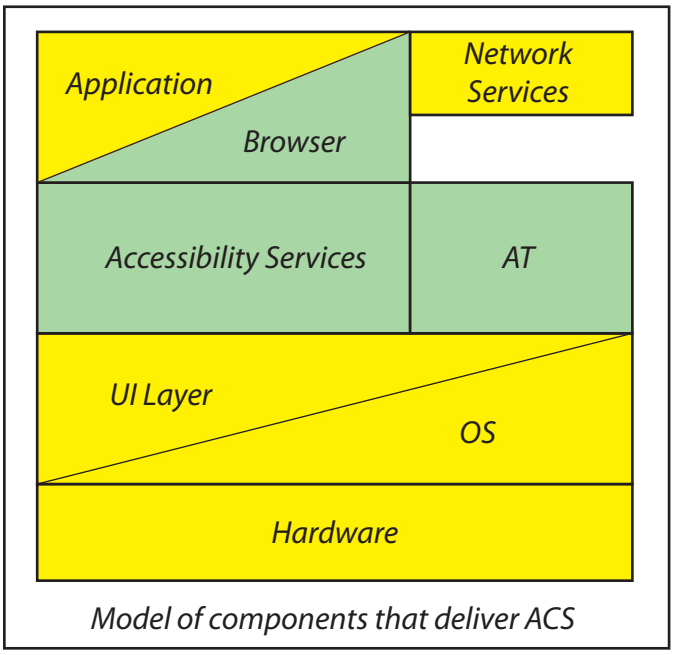
ITI looks forward to the opportunity of providing in-person demonstrations to illustrate some of the examples contained in these comments. In the meantime, we welcome any inquiries regarding the views expressed herein. Please direct any questions to the undersigned.

Respectfully,

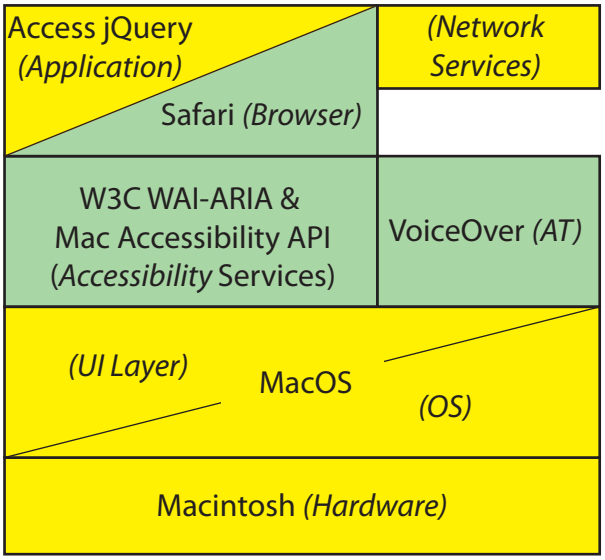
Ken J. Salaets
Director

ATTACHMENTS: Model of ICT Accessibility Components – Visualization
Model of ICT Accessibility Components – Textual Description

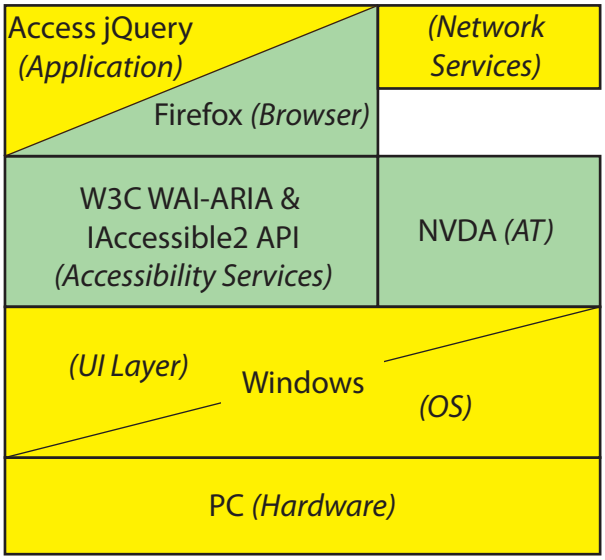
DRAFT one-page picture for ITI FCC ex-parte meeting, 8June11



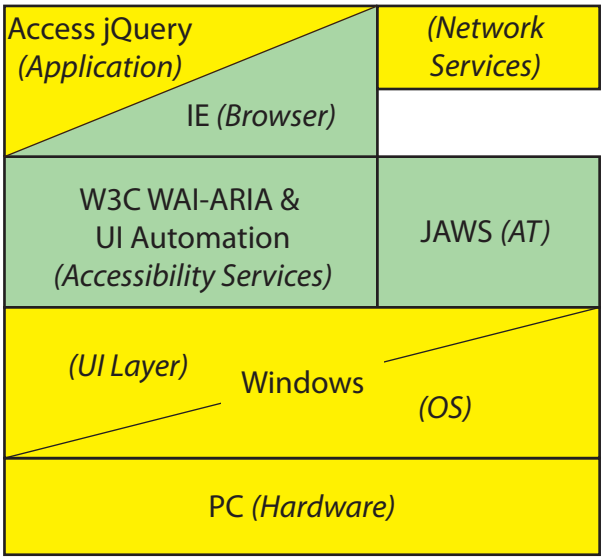
Model of components that deliver ACS



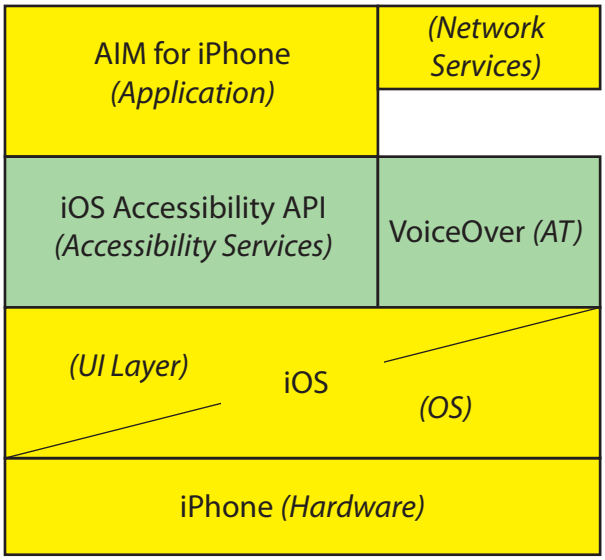
Example #1: built-in AT on Macintosh



Example #2: nominal cost AT on Windows



Example #3: compatibility with AT on Windows



Example #4: built-in AT on iPhone

Textual Description of the Document Entitled "Model of ICT Accessibility Components"

The graphic consists of 5 pictures, each a variant of the first picture at the top of the page, with the remaining four pictures in a 2x2 grid underneath the first centered at the top. The first/top picture shows the 5 "components of ACS" as described by the FCC in the NPRM paragraph 15, along with the 3 additional components proposed by ITI. The 5 components noted by the FCC have a light yellow background, while the 3 components proposed by ITI have a light green background. The components/boxes are: Hardware (on the bottom of a stack), OS and UI Layer (shown sharing a split box on top of Hardware), Accessibility Services and AT as separate boxes sitting beside each other on top of OS and UI Layer, Application and Browser (shown sharing a split box) on top of Accessibility Services, and finally somewhat off to the side connected off of Application and Browser is Network Services. This picture/stack is labeled "Model of components that deliver ACS". Each of the other 4 pictures is this same stack, but labeled with the names of specific products/technologies, each as appropriate to the Example captioning them. Unlike the rest, this first picture/stack has a further rectangle around it, to better illustrate that it is the model for the rest. Also, all of the text is in italics. For the remaining pictures/stacks, the model text is repeated, in italics and parenthesis, after each product/technology name example.

The second picture/stack is on the center left, captioned "Example 1: built-in AT on the Macintosh". Here hardware is "Macintosh (Hardware)". Above that is the split box with "MacOS" in the center, and with "(UI Layer)" and "(OS)" in the upper left/lower right of that box. Above that to the left is "W3C WAI-ARIA and Mac OS X Accessibility Protocol (Accessibility Services)", and to the right "VoiceOver (AT)". Above Accessibility Services is the split box with "AOL ACS (Application)" in the upper left and "Safari (Browser)" in the lower right. Connected to this box, off to the side, is "(Network Services)". This picture/stack is used to support a demonstration of jQuery widgets (as part of an "AOL ACS application") being made accessible with VoiceOver on a Macintosh, by way of the Safari Browser utilizing WAI-ARIA from the jQuery app and the Mac Accessibility API providing the info to VoiceOver. This also illustrates ITI's first bullet point on page 5 in our public ACS NPRM comments "Built-In AT".

The third picture/stack is on the center right, captioned "Example #2: nominal cost AT on Windows". The bottom box is "PC (Hardware)". Above that is the split box "Windows" with "(UI Layer)" and "(OS)" in upper left/lower right of that same box. Above that is "W3C WAI-ARIA & IAccessible2 API (Accessibility Services)", similar as to what is in Example #1 (and specifically, the first part - "W3C WAI-ARIA" - is the same). To the right of that is "NVDA (AT)". Above Accessibility Services is a split box with "AOL ACS (Application)" - as before – and "Firefox (Browser)". Finally, "(Network Services)" off to the side, as before. This picture/stack is used to support a demonstration of the same jQuery widgets as before, but being made accessible with NVDA on Windows. This is a Windows version of illustrating ITI's

second bullet on page 5 in our public ACS NPROM comments “Third-party AT available for free”.

The four picture/stack is on the lower left, captioned “Example #3: compatibility with AT on Windows”. It is very similar to the third picture/second example. Only replace “NVDA (AT)” with “JAWS (AT)”, and replace “Firefox (Browser)” with “IE (Browser)”, and “IAccessible2 API” with “UI Automation”. This picture/stack is used to support a demonstration of the same jQuery widgets as in the last two demos, but being made with JAWS on Windows. This also illustrates ITI's third bullet point on page 6 in our ACS NPRM comments “Third-party AT available for a fee.” These three examples together demonstrate our key points:

- The same ACS software application – unchanged – may be accessible via built-in AT, nominal cost AT, or 3rd party commercial AT *without doing anything differently, simply by virtue of the ICT ecosystem of the end-user's platform*

- The ACS software application has no control over what browser, platform, hardware, or AT is being used by the end user

- What the ACS software application can – and should – do is support the appropriate accessibility services. In this case, W3C WAI-ARIA.

The final example is on the lower right, captioned “Example #4: built-in AT on an iPhone." This picture/stack shows that the model applies to native applications and to the mobile environment, just as well as it does to desktop computers and browsers. The bottom box is “iPhone (Hardware)”. Above that is the split box “iOS” with “(UI Layer)” and “(OS)” in upper left/lower right of that same box. Above that is “UI Accessibility (Accessibility Services).” To the right of that is “VoiceOver (AT)”. Above Accessibility Services is a box with “AIM for iPhone (Application)”. Finally, “(Network Services)” off to the side, as ever.